

**Naive Bayes Text Classification**

**CS 585 Homework-1**



Name- Swapnil Sopan Gaikwad

Online Student (India)

CWID - A20377532

You can run the code like:

python NaiveBayes.py data/aclImdb 1.0

### Classification and Evaluation (40 Points)

self.P\_positive = P(+) = 0.5

self.P\_negative = P(-) = 0.5

size of self.count\_positive = 252192

size of self.count\_negative = 252192

self.vocab\_len = 252192

self.total\_positive\_words = 2958696

self.total\_negative\_words = 2885722

|  |  |  |  |
| --- | --- | --- | --- |
| **ALPHA** | **Accuracy** | **self.deno\_pos** | **self.deno.neg** |
| 0.1 | 0.81688 | 2983915.2 | 2910941.2 |
| 0.5 | 0.82668 | 3084792.0 | 3011818.0 |
| 1.0 | 0.8304 | 3210888.0 | 3137914.0 |
| 5.0 | 0.83632 | 4219656.0 | 4146682.0 |
| 10.0 | 0.83712 | 5480616.0 | 5407642.0 |

### Probability Prediction (20 Points)

The probability estimated for the first 10 reviews in the test data.

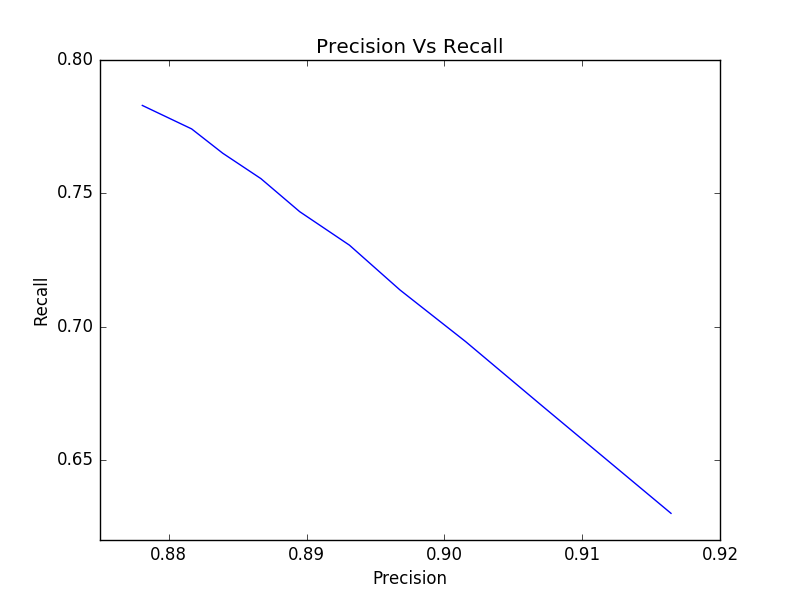
ALPHA = 1.0

|  |  |  |  |
| --- | --- | --- | --- |
| **Index of test.X[i]** | **test.Y[i]** | **predicted\_prob\_positive** | **predicted\_prob\_positive** |
| 0 | -1.0 | 0.9365132665902932 | 0.06348673340965141 |
| 1 | -1.0 | 1.4294347203216804e-07 | 0.9999998570565469 |
| 2 | -1.0 | 4.388356852437946e-17 | 1.0 |
| 3 | -1.0 | 4.078027965195846e-13 | 0.9999999999995453 |
| 4 | -1.0 | 4.775228165409764e-13 | 0.9999999999995453 |
| 5 | -1.0 | 8.66332518621793e-08 | 0.9999999133667681 |
| 6 | -1.0 | 3.3444380516436173e-08 | 0.9999999665556066 |
| 7 | 1.0 | 0.020877247067161847 | 0.9791227529328611 |
| 8 | -1.0 | 0.9794966557702657 | 0.020503344229679848 |
| 9 | -1.0 | 1.4350093016703943e-05 | 0.9999856499070662 |

### 

### Precision and Recall (20 Points)

Graph precision vs. recall for the positive and negative classes by varying the threshold.



Precision α (1/Recall)

i.e. there is **inverse relationship** between Precision and Recall.

### Features (20 points)

The 20 most positive and 20 most negative words in the vocabulary sorted by their weight according to model.

negative\_word\_weight = log(P(-|w)) - log(P(+|w))  
  
positive\_word\_weight = log(P(+|w)) - log(P(-|w))

**Top +ve words** = [('edie', 4.395851321341416), ('gundam', 4.320816135398502), ('antwone', 4.10414509858991), ('yokai', 3.8482117244527085), ('/>8/10', 3.8482117244527085), ('gunga', 3.827158315254877), ('/>7/10', 3.827158315254877), ('/>10/10', 3.805652110033913), ('din', 3.7836732033151375), ('gypo', 3.7836732033151375), ('othello', 3.7382108292383798), ('7/10.', 3.6145968732712035), ('tsui', 3.560529652000927), ('paulie', 3.546543410026187), ('blandings', 3.532358775034231), ('goldsworthy', 3.4735182750112976), ('kells', 3.442746616344545), ('gino', 3.442746616344545), ('/>9/10', 3.442746616344545), ('harilal', 3.410997918029963)]

**Top -ve words** = [('/>4/10', 4.0660405542897315), ('seagal', 4.057229924607578), ('2/10', 3.914809584565809), ('boll', 3.9045530843986196), ('uwe', 3.894190297363073), ('\*1/2', 3.8516306829442772), ('unwatchable.', 3.829651776225502), ('thunderbirds', 3.76065890473855), ('/>3/10', 3.736561353159491), ('gamera', 3.736561353159491), ('4/10', 3.673647527748921), ('wayans', 3.633907199099406), ('awful!', 3.5783373479445952), ('slater', 3.488725189254909), ('/>avoid', 3.488725189254909), ('segal', 3.4569764909403276), ('drivel.', 3.4569764909403276), ('tashan', 3.4569764909403276), ('kareena', 3.424186668117338), ('aztec', 3.424186668117338)]

**Resources:**

[1] [http://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.sparse.csr\_matrix.html - scipy.sparse.csr\_matrix](http://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.sparse.csr_matrix.html%20-%20scipy.sparse.csr_matrix)

[2] <https://en.wikipedia.org/wiki/List_of_logarithmic_identities>

[3] <https://stats.stackexchange.com/questions/105602/example-of-how-the-log-sum-%20exp-trick-works-in-naive-bayes>